UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/026,266	12/21/2001	Michael L. Fraenkel	RSW920010208US1	7861

7590 09/28/2004

A. Bruce Clay
IBM Corporation T81/503
PO Box 12195
Research Triangle Park, NC 27709

EXAMINER	
STEELMAN, MARY J	

ART UNIT	PAPER NUMBER
2122	

DATE MAILED: 09/28/2004

2

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	10/026,266	FRAENKEL ET AL.	
	Examiner	Art Unit	
	Mary J. Steelman	2122	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *21 December 2001*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-20* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-20* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☒ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *21 December 2001* is/are: a)☐ accepted or b)☒ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date *12/21/2001*.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-20 are pending.

Drawings

2. The drawings are objected to as failing to comply with 37 CFR 1.84(p)(5) because they do not include the following reference sign(s) mentioned in the description: FIG. 1, #100. Corrected drawing sheets in compliance with 37 CFR 1.121(d) are required in reply to the Office action to avoid abandonment of the application. Any amended replacement drawing sheet should include all of the figures appearing on the immediate prior version of the sheet, even if only one figure is being amended. The replacement sheet(s) should be labeled "Replacement Sheet" in the page header (as per 37 CFR 1.84(c)) so as not to obstruct any portion of the drawing figures. If the changes are not accepted by the examiner, the applicant will be notified and informed of any required corrective action in the next Office action. The objection to the drawings will not be held in abeyance.

Specification

3. The use of the trademark JAVA has been noted in this application. It should be capitalized wherever it appears and be accompanied by the generic terminology.

Although the use of trademarks is permissible in patent applications, the proprietary nature of the marks should be respected and every effort made to prevent their use in any manner which might adversely affect their validity as trademarks.

Claim Rejections - 35 USC § 101

4. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

the claimed invention is directed to non-statutory subject matter.

Claims 1-3 are rejected under 35 U.S.C. 101 because they are directed towards a "class loader", which is a program per se, and thus non-statutory. This may be cured by amending claims 1-3 to recite "A custom class loader apparatus configured to..."

Claim Rejections - 35 USC § 112

5. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

6. Claim 3 is rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

7.35.01 Trademark or Trade Name as a Limitation in the Claim

Claim 3 contains the trademark/trade name JAVA. Where a trademark or trade name is used in a claim as a limitation to identify or describe a particular material or product, the claim does not comply with the requirements of 35 U.S.C. 112, second paragraph. See Ex parte Simpson, 218 USPQ 1020 (Bd. App. 1982). The claim scope is uncertain since the trademark or trade name cannot be used properly to identify any particular material or product. A trademark or trade name is used to identify a source of goods, and not the goods themselves. Thus, a trademark or trade name does not identify or describe the goods associated with the trademark or trade name. In the present case,

the trademark/trade name is used to identify/describe virtual machine class loader and, accordingly, the identification/description is indefinite.

7. Claim 17 recites the limitation "each parent class loader..." in line 3. There is insufficient antecedent basis for this limitation in the claim.

Claim Rejections - 35 USC § 103

8. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

9. Claims 1-20 are rejected under 35 U.S.C. 103(a) as being unpatentable over US Patent Application Publication 2004/0015936 A1 to Susaria et al.

Per claim 1:

-class loading logic configured to specifically and dynamically locate, define and load a class specified by name;

([0056], "Each module in the application may be associated with its own class loader...")

-a list of peer class loaders arranged in accordance with the associated dependency specification and, list generation logic configured to generate said list when said specified class has been replaced or when said dependency specification has been modified;

([0056], "The class loader module may include a hierarchical stack of class loaders. Each class loader may have one parent class loader and zero or more child class loaders...")

-a flag indicating whether said class has been replaced;

([0059], ...application may include a dirty class monitor...")

-deference logic configured to defer said location, definition and loading of said specified class to said peer class loaders in said list.

(0057, "...class loaders that are each configured to load one or more classes for the application when invoked." Class loaders in the hierarchy (peers) contain the logic to load specific classes.)

Susaria disclosed (Abstract) a dynamic class loader. The class loader module may include a hierarchical stack of class loaders. Each module in the application may be associated with its own class loader. Each class loader may be responsible for loading one or more classes. When a class is changed. The changed class may be detected. The concerned class loader can be replaced. The class loaders for all classes that depend on the changed class may also be replaced. The replaced class loaders may then reload the affected classes.

Susaria failed to specify a list of peer class loaders. However, a hierarchical stack of class loaders is a data structure with a similar effect. Peer class loaders can be identified. Susaria failed to specify "list generation logic", generated when said specific class has been replaced or when said dependency specification has been modified. Although Susaria failed to generate a "list", he did disclose that when a class was replaced / modified, that dependent classes are provided with appropriate class loaders. Susaria failed to specify "a flag comprises a dirty bit" to indicate a class has been replaced. However, he did provide a dirty monitor that noted a changed class [0059], which provides the same information. When a "Dirty Class Monitor" notifies that a replacement loader is needed, the "Application Class Loader Controller"

finds a peer loader to load the modified class and suitable loaders for any dependent classes. [0059], "The application may detect that a class has been changed...may include a dirty class monitor that may monitor classes used by the application and detect when any of the classes have been changed. The class loader for the class may be replaced with a new version of the class loader..."

Therefore, it would have been obvious, to one of ordinary skill in the art, to consider the effects of Susaria's invention to be obvious over Applicant's claimed invention. When a change is detected, a new class loader (peer) is used to load the changed class.

Per claim 2:

-said flag comprises a dirty bit.

(FIG. 8 & [0142-0143], "Dirty Class Monitor",[0059], "The application may detect that a class has been changed...a dirty class monitor...")

Per claim 3:

-custom class loader conforms to the specification of a JAVA™ version 1.2 delegation-style custom class loader.

([0081], "Follow a delegation mechanism that is a modification of the mechanism described in JDK, version 1.2.")

Per claim 4:

-receiving a request to load a specified class;

(FIG. 7, [0100], "...it may forward the "load class" request to the class loader controller", [0106], "load class requests")

-determining whether said specified class has been replaced;

([0088], "If a class changes, all the classes that use this class may be reloaded as well...", [0100], "The class loader controller then may determine which class loader is supposed to load the class.")

-if it is determined that said specified class has been replaced, constructing a new instance of the class loader and generating a list of peer class loaders to which location, definition and loading of said specified class are to be deferred in accordance with a dependency specification in the virtual machine;

([0100], "Any notification for a class change may also come to the class loader controller so that it can recreate the concerned class loaders.", [0108], "Receiving a notification from the replacement logic of the application class loader controller when a class changes.", [0110], ...application class loader controller may also be responsible for dispatching the "load class" requests to the appropriate class loader.", [0141], "The class loader controller may then replace the class loader with the new class loader. If there are one or more classes that depend on the class to be reloaded, the class loaders responsible for reloading the dependent classes may be located and replaced as well..." Also see FIG. 7, #308 regarding dependencies.)

-deferring said location, definition and loading to said peer class loaders in said list.

(0141], "the class loader controller may invoke each of the necessary class loaders to reload the classes that need to be reloaded in response to the change in the class.")

Per claim 5:

-determining step comprises checking a dirty bit in the class loader.

([0140], "dirty class monitor that may monitor classes used by the application and detect when any of the classes have been changed.")

Per claim 6:

-traversing each peer class loader in said dependency specification;

(FIG. 7, [0138], "application may include a class loader module that may include a hierarchical stack of class loaders that are each configured to load one or more classes for the application when invoked.", [0141], "The class loader controller may then locate the class loader responsible for loading the class in the hierarchical stack of class loaders.")

-adding a reference for each said traversed peer class loader to said list.

Per claim 7:

-dependency specification comprises a tree of nodes, each said node encapsulating a reference to a dependency of said specified class, one of said nodes encapsulating a reference to said specified class.

([0141], "The class loader controller may then replace the class loader with the new class loader. If there are one or more classes that depend on the class to be reloaded, the class loaders responsible for reloading the dependent classes may be located and replaced as well.")

Per claim 8:

-beginning with said one node encapsulating a reference to said specified class, traversing each node in said dependency specification using a depth-first traversal strategy until encountering either a leaf node or a node encapsulating a reference to a dependency already referenced in said list;

-responsive to said encountering, traversing each node in said dependency specification using a breadth-first traversal strategy until encountering said node encapsulating said reference to said specified class;

-adding a reference for each traversed node to said list.

(See response to claim 7 above. Class loaders are located in the hierarchical stack of class loaders. In some cases one loader may load multiple classes. In some cases there may be a specific loader associated with a specific class. Classes that are dependent upon a changed class, may also need to be reloaded with a modified class loader.

Per claim 9:

-adding at least one reference to a peer class loader to said list based upon a corresponding reference stored in a list of peer class loaders identified in one of said traversed peer class loaders.

([0145], "Helper (utility) classes in one module may be symbolically referenced by other module classes...helper classes may be loaded by the same class loader (peer class loader)...:

Per claim 10:

-setting said dirty bit responsive to said specified class being replaced.

([0142-0143], "Dirty Class Monitor FIG. 8 illustrates a dirty class monitor...Tasks related to the dirty class monitor may include, but are not limited to, registration and notification.")

Per claim 11:

-setting each dirty bit in each peer class loader referenced in said list responsive to said specified class being replaced.

([0142-0143] & FIG. 8)

Per claim 12:

(See limitation addressed in claim 4 above.)

Per claim 13:

(See limitation addressed in claim 5 above.)

Per claim 14:

(See limitations addressed in claim 6 above.)

Per claim 15:

(See limitations addressed in claim 7 above.)

Per claim 16:

(See limitations addressed in claim 8 above.)

Per claim 17:

-traversing each parent class loader associated with the class loader through to a primordial class loader;

([0088], "if the system class loader is assumed to be at the highest level (primordial class loader), then a change in a system class may trigger class reloading in all the lower levels...")

-adding a reference for each said traversed parent class loader to said list.

([0056], "The class loader module may include a hierarchical stack of class loaders. Each class loader may have one parent class loader and zero or more child class loaders. Each module in the application may be associated with its own class loader...")

Per claim 18:

-adding at least one reference to a parent class loader to said list based upon a corresponding reference stored in a list of parent class loaders identified in one of said traversed parent class loaders.

([0056], "The class loader module may include a hierarchical stack of class loaders. Each class loader may have one parent class loader and zero or more child class loaders. Each module in the application may be associated with its own class loader...")

Per claim 19:

(See limitations addressed in claim 10 above.)

Per claim 20:

(See limitations addressed in claim 1 above.)

Conclusion

10. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Mary Steelman, whose telephone number is (703) 305-4564. The examiner can normally be reached Monday through Thursday, from 7:00 AM to 5:30 PM If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Dam can be reached on (703) 305-4552. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

After October 25, 2004, examiner can be reached at new telephone number (571) 272-3704. Supervisor, Tuan Q. Dam can be reached at (571) 272-3694.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Mary Steelman

09/14/2004

TUAN DAM
SUPERVISORY PATENT EXAMINER